AMENDMENTS TO THE SPECIFICATION

Please replace paragraph [0001] with the following paragraph:

[0001] The subject matter of this application is related to the subject matter in a co-pending non-provisional application by the same inventors as the instant application entitled, "Dynamically Configuring Selected Methods for Instrument-Based Profiling at Application Run-Time," having serial number 10/654,522, and a filing date of 02 September 2003 (Attorney Docket No. SUN-P9153-SPL). The subject matter of this application is also related to the subject matter in a co-pending non-provisional application by the same inventors as the instant application entitled, "Method and Apparatus for Performing Time Measurements During Instrumentation-Based Profiling," having serial number 10/666,515~~TO BE ASSIGNED~~, and filing date ~~TO BE ASSIGNED~~18 September 2003 (Attorney Docket No. SUN-P9376-SPL).

Please replace paragraph [0003] with the following paragraph

[0003] The growing size and complexity of modern software applications is increasing the demand for tools that automate the process of collecting data about the dynamic behavior of programs, thereby allowing developers to identify performance bottlenecks in their applications. The process of automatic collection and presentation of data characterizing performance of running programs is called "profiling." For an object-oriented language such as the JAVA™ (hereinafter "Java") programming language, that features automatic memory management, builtin multithreading and thread synchronization mechanisms, etc., several forms of profiling are useful in practice. (Note that Java is a trademark or a registered trademark of Sun Microsystems, Inc. in the United States and other countries.) They are distinguished by the type of data collected: CPU profiling determines how much time the program spends

executing various parts of its code; memory profiling determines the number, types and lifetime of objects that the program allocates; monitor profiling determines congested monitors, etc. For applications built according to higher-level standards, for example Enterprise Java applications, specialized, high-level kinds of profiling exist, for example measuring the number of transactions passing through the system in a second.

Please replace paragraph [0006] with the following paragraph:

[0006] When we consider programs executed on top of a virtual machine (VM), as it is the case with the JAVA™ platform, we have to mention another alternative to code instrumentation: VM-generated events, or "VM hooks." The VM itself can be instrumented to generate events such as method entry/exit, object allocation, monitor enter, etc. This is done essentially by placing calls to user-suppliable functions in the relevant places in the VM code, for example in the interpreter code executed upon method entry. Some events that are important when profiling a Java application, for example a garbage collection event, cannot be generated using bytecode instrumentation at all. However, for most of the other events, in particular for method entry/exit and object allocation, it has been found over time that their support inside a ~~JVM~~ JVM™ (hereinafter "JVM") complicates the latter, sometimes requiring a significant effort from developers, and at run time may cost more than equivalent bytecode instrumentation. (Note that JVM is a trademark or a registered trademark of Sun Microsystems, Inc. in the United States and other countries.) This is true at least for VMs intended to be used for general-purpose desktop and server applications, in contrast with those used in cell phones and smaller devices. As a result, it has been recently decided by the expert group established to define a new JVM profiling API, that in the forthcoming specification, many of the VM-generated events, including method entry/exit and object allocation, will be optional and not required to be

supported by all conforming JVMs (see JSR 163—Java Platform Profiling Architecture~~, http://www.jcp.org/jsr/detail/163.jsp~~, 2002). Bytecode instrumentation is the recommended mechanism for their generation.